

ABSTRACT

IAM CORE
(Identity Access Management)
Documentation of JAVA Fundamentals.

Sabari Nilash Kurapati
M.sc(SE)



Table of Contents:

Subject description

Subject analysis

- Major features

- Application Feasibility

- Data description

- Expected results

- Algorithms study

- Scope of the application (limits, evolutions)

Conception

- Chosen algorithm

- Data structures

- Global application flow

- Global schema and major features schema

Console operations description

- <One section by operation>

Configuration instructions

Commented Screenshots

Bibliography



Subject description

This project has a good covering basic things, like creating and using the Java Developer Kit APIs, but also to persist data in databases, or executing an application on a Java application Server.

The subject of Identity Management System's main goal is to manage users of an Information System. As many basic concepts, it can be much improved, especially when you want to bring security to this management. The application will be able to :

- Access, create, Search, delete and modify user information

- Persist users data in a database (or in an XML File)

- Be robust, capable of good performance



Subject analysis

Major features

The IAMProject has 4 major components.

- Login Authentication: A User should be authenticated before been able to work and manage the Identities.
- Manage Identities: the user can be able to
 - Create
 - Search.
 - Update
 - Delete

The Identities and the users are stored as application database. DERBY used as the Database engine for managing the entities.



Data description

There are two kinds of data related with this application, which consist of user data and identity data:

1. Identity data – which include Uid, Display name, Email.

- DisplayName:

 - Type: string

- Uid:

 - Type: string

- Email

 - Type: string

2. User data – which include user and password

- Username is a string combine with name as credential (for authentication)

- Password is a string combine with name as credential (for authentication)

Expected results

Users with administrative privileges can create, delete and modify identities.
User provided information can be stored in database.
User can search Identities using his User ID.

Scope of the application

The application is limited to only manage Identities and not to manage the Users. This is of course one of the first steps to be improved in a next revision. Also, the Identities are only going to be managed on a database only scope of User Authentication, Creation, Deletion, Updating of the identities and for now file storage mode for Identities is not implemented, so the user must have an installed Database.

Limitations

- User roles are not defined.
- Services are not secured.
- Password is exposed in payload sent for authentication.
- Need to go back to authentication page for every operation.
- Not having forget user name and password option.

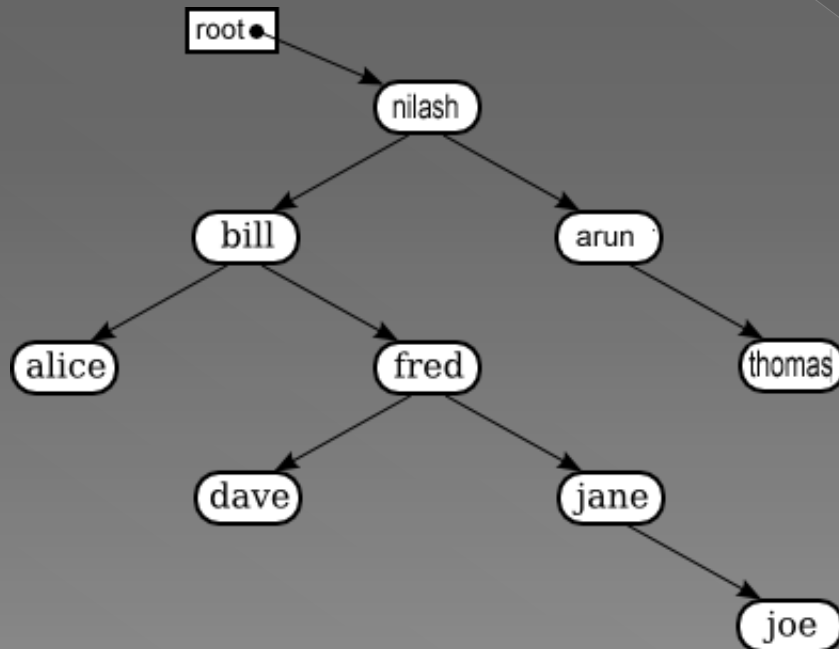


Conception

Chosen algorithm

Searching Algorithm:

Comparison based sorting techniques like quick sort, merge sort has $O(n \log n)$ complexity while non comparison sort techniques like counting sort has $O(n)$ complexity. Also, binary search is done on a sorted array.



Data structures

Identities

Identities are basically the entities that are going to be managed, within the application we should be able to create, update and delete these. All identities are going to be stored in a Derby database.

As per the requirements an identity can have:

- UId: User Id, a unique id combination of username and email as a Primary Key in the Identity table.
- Display Name: represented as a string.
- Email: represented as a string also.

Users

Users are not supposed to be managed by the application, for this reason a decision was made that user entities will be stored in a text file.

A user will have a name and a password; both are stored as string files.



Business Logic Description

POJO Classes

- IdentityManagement - Class used for Identity table
- UserDetails - Class used for user table
- UserManagementVO - Class used to wrap IdentityManagement and UserDetails class and send object to UI.

ReadPropertiesFile - Class used to read Property file objects and store them In Map.

Auth - Class used for login authentication.

IdentityConstants - Class contains constants used in application.

DataManagementDAOImpl - Class get Database object.

GetJDBCConnection - Class used to connect database.



PrepareSQLQueries - Class contain functions used to prepare SQL queries.

DataManagementHandlerImpl - Class used to handle DAO object.

DataManagementDTO - Class used to transform data object to UI object and vice versa.

IdentityException - Class is main exception class of application and it extend Exception class.

TestFileOperations – Class contains all test methods of application.

Testlauncher – Class is used to launch TestFileOperations class.

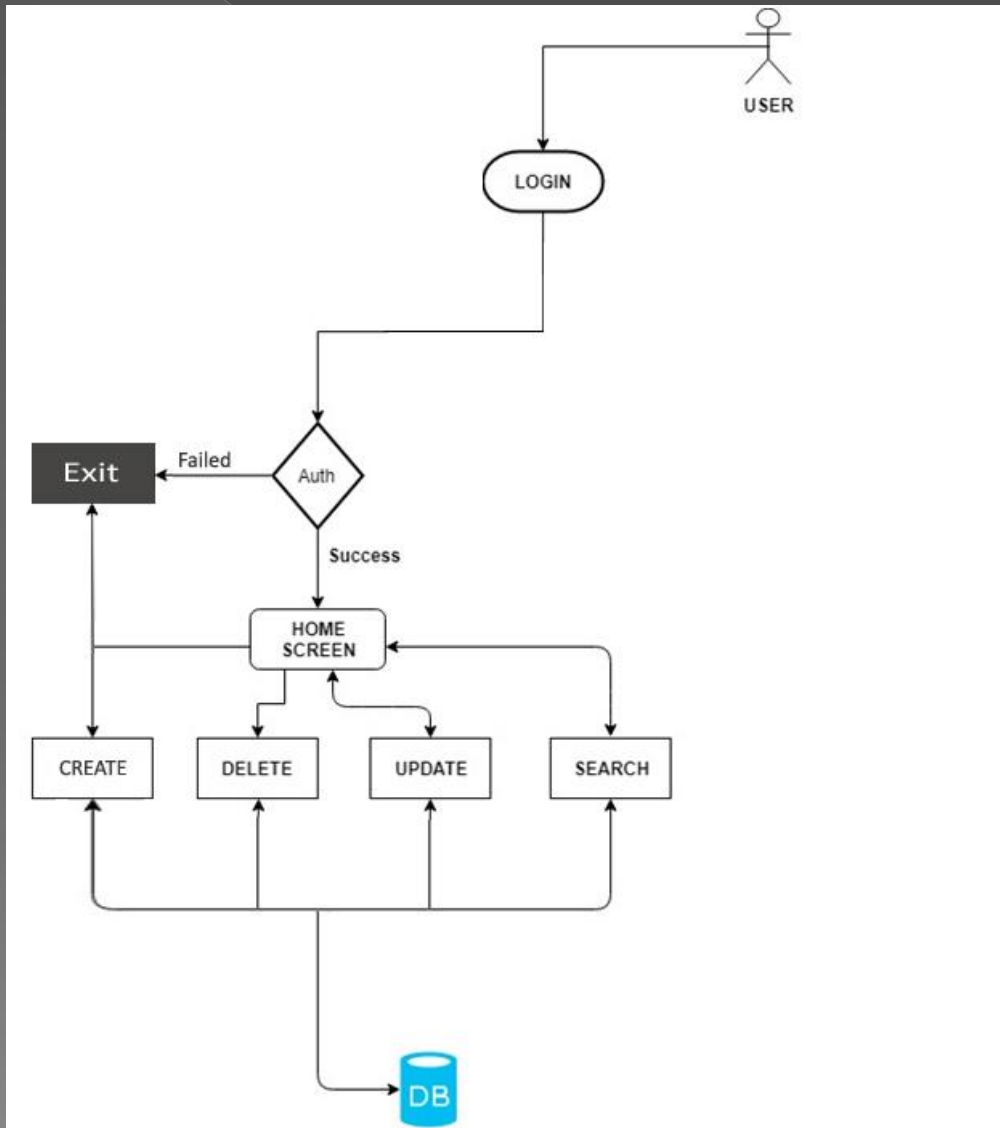
Configuration – Class is used to read IncidentManagement.properties file.

Logger - Class is used to generate logfile name log4j-application.log .

IdentityException - Class is main exception class of application and it extend Exception class.



Global application flow



Global schema and major features schema

Identity Management Schema

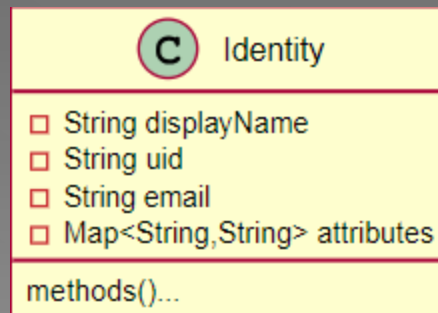
Means to read and store information about identities. This feature can be achieved in several ways (these means are listed from the simplest to the hardest).

The minimal thing is to provide a set of API to act on identities data
You also can do it through a SWT application thanks the GUI builder in Eclipse see : [Eclipse gui builder](#)

If you are used to using servlets, JSP and more generally web interface, you can do a little java web application.

Data can be persisted by reading/writing data from/in a raw file, an xml file or a database using a JDBC connector.

As an example of an identity definition, you can look at the schema hereafter:



Console operations description

Authentication

UserName: User should enter his registered username, Otherwise user cannot login using wrong user name.

Password: User should enter his registered password, Otherwise user cannot login using wrong password.

User Operations

Create: User able to create new identity in database with his Name, Email and UID.

Search: User able to search his identity in database with his Name and Email.

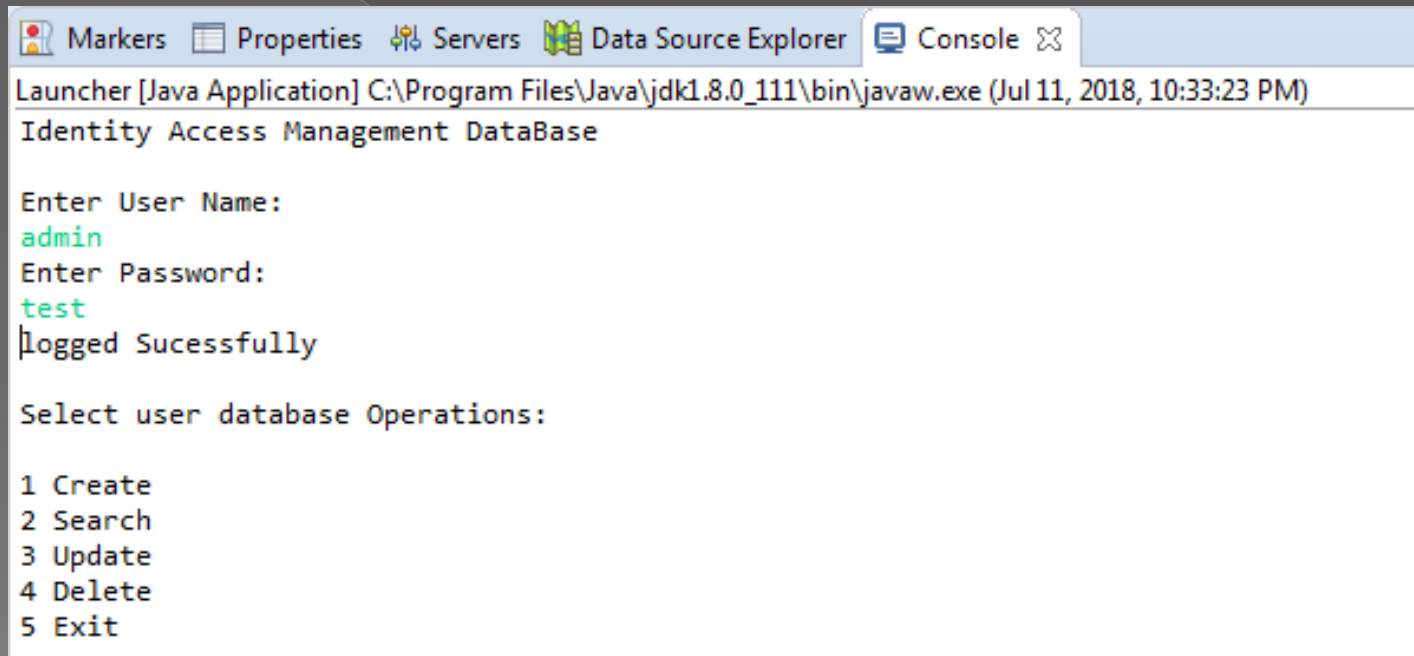
Update: User able to update his existing identity data in database with his UID.



Delete: User able to delete his identity in database with his Name and Email.

Exit: User can able to exit from session.

Below figure shows the console operations:



The screenshot shows a Java application window titled "Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:33:23 PM)". The window has tabs for "Markers", "Properties", "Servers", "Data Source Explorer", and "Console". The "Console" tab is active, displaying the following text:

```
Identity Access Management DataBase

Enter User Name:
admin
Enter Password:
test
logged Sucessfully

Select user database Operations:

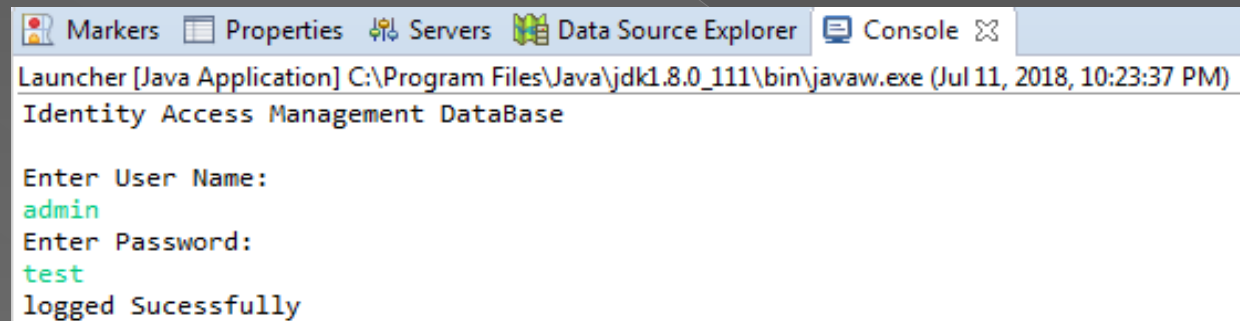
1 Create
2 Search
3 Update
4 Delete
5 Exit
```

Commented Screenshots

The implementation respected this diagram to fulfill the requirements. A description of this flow is done with Console Operations screenshots in the following pages:

Welcome Title and Authentication Form

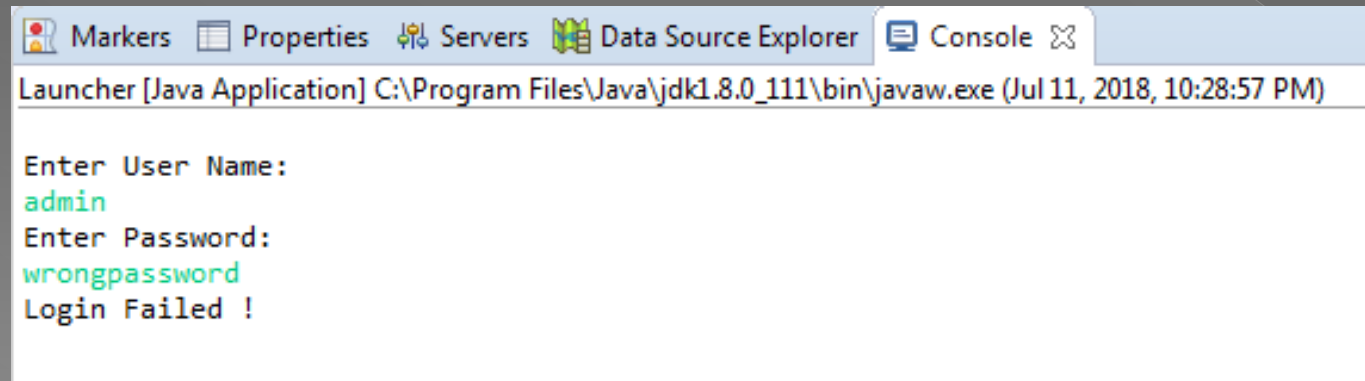
Correct Password



```
Markers Properties Servers Data Source Explorer Console
Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:23:37 PM)
Identity Access Management DataBase

Enter User Name:
admin
Enter Password:
test
logged Sucessfully
```

Wrong Password

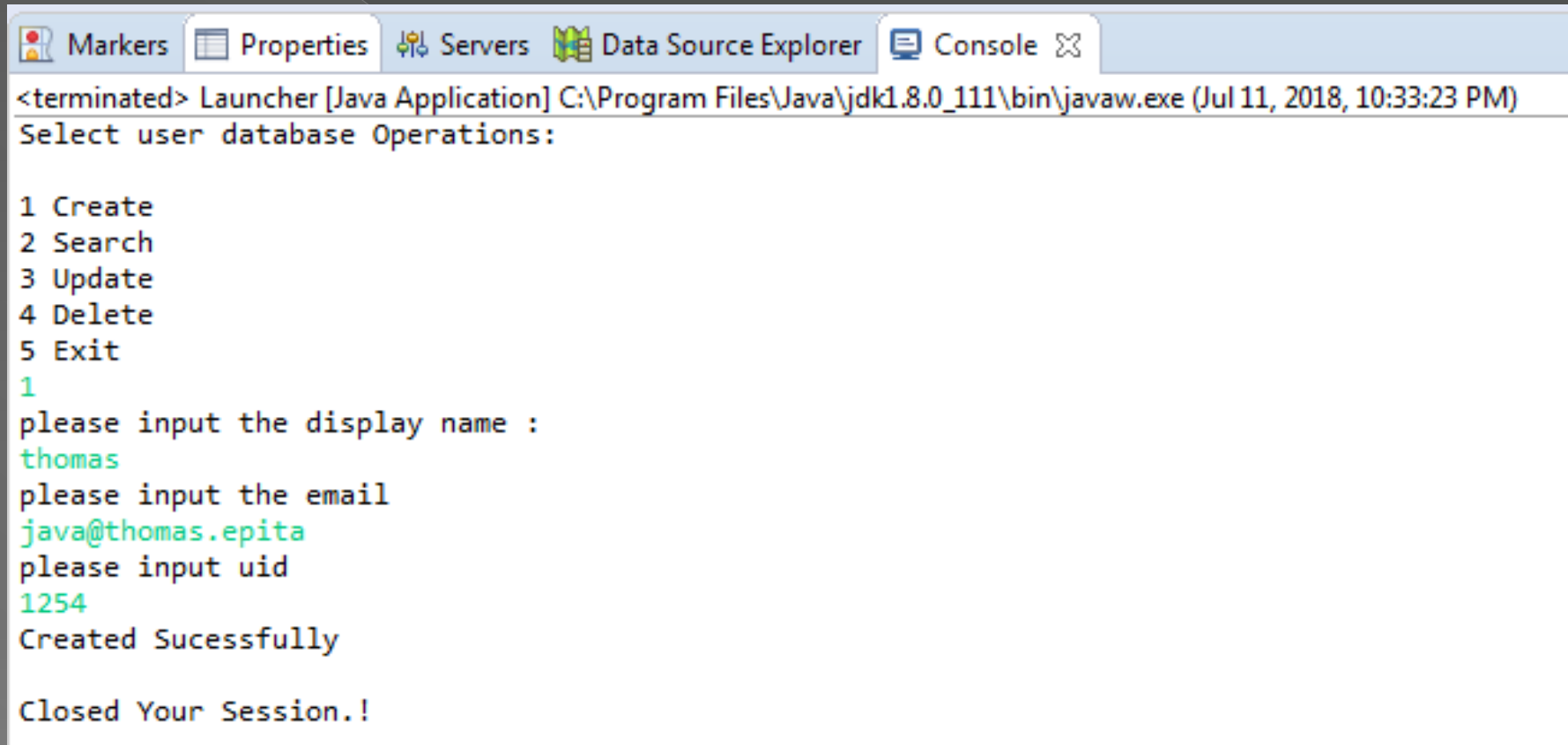


```
Markers Properties Servers Data Source Explorer Console
Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:28:57 PM)

Enter User Name:
admin
Enter Password:
wrongpassword
Login Failed !
```

Create Operation

the user is required to input the Display Name, Email and User ID. The Identity will be created using user inputs on the console.



The screenshot shows a Java application window titled "Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:33:23 PM)". The window has several tabs: "Markers", "Properties", "Servers", "Data Source Explorer", and "Console". The "Console" tab is active, displaying the following text:

```
<terminated> Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:33:23 PM)
Select user database Operations:

1 Create
2 Search
3 Update
4 Delete
5 Exit
1
please input the display name :
thomas
please input the email
java@thomas.epita
please input uid
1254
Created Sucessfully

Closed Your Session.!
```

The user input "thomas" for the display name, "java@thomas.epita" for the email, and "1254" for the user ID are shown in green text. The application successfully creates the user and closes the session.

Search Operation

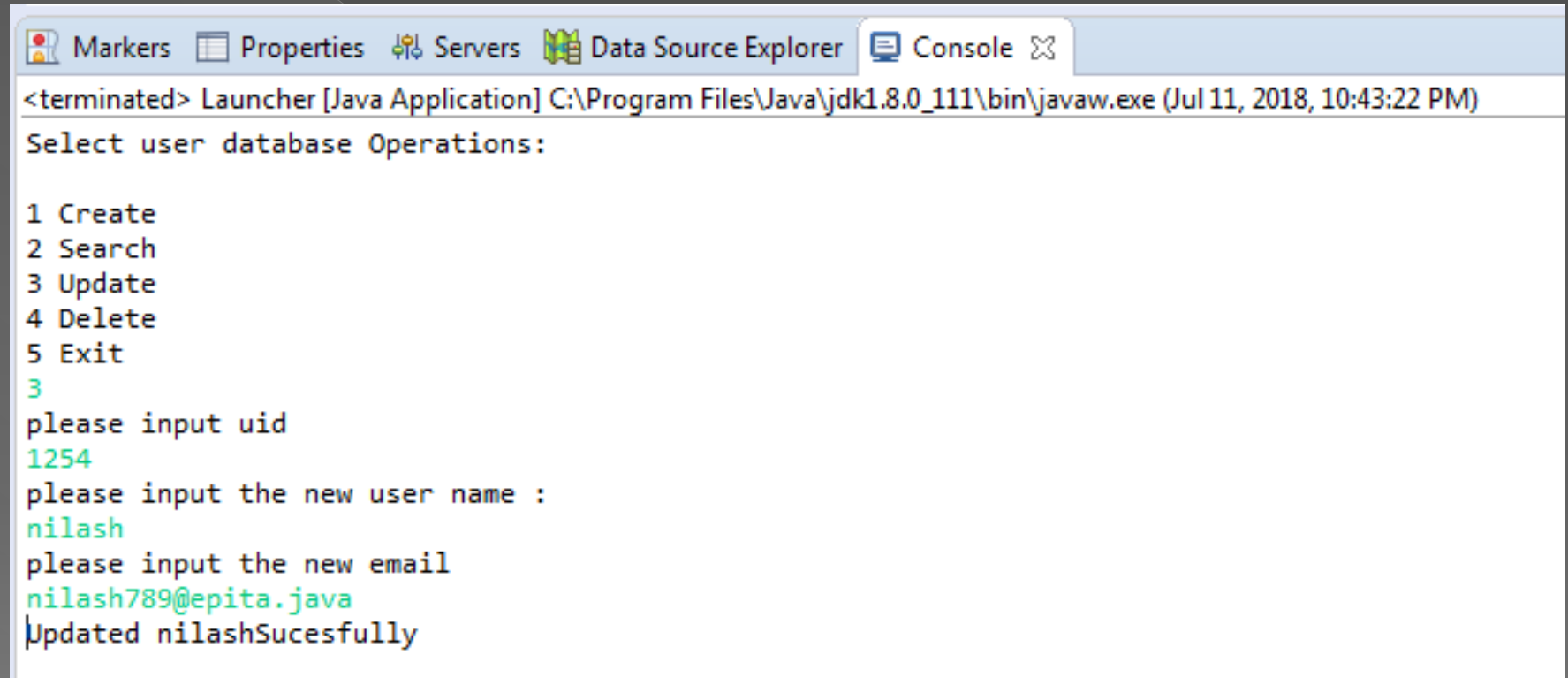
the user is required to input the Display Name and Email. The search operation runs in the background and get the output from database based on the user inputs on the console.



```
Markers Properties Servers Data Source Explorer Console
<terminated> Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:40:35 PM)
Select user database Operations:
1 Create
2 Search
3 Update
4 Delete
5 Exit
2
Enter criteria
please input the criterion for display name :
thomas
please input the criterion for email
thomas
Result:
1 - Identity [displayName=thomas, uid=788, email=thomas@java.com, attributes=null]
2 - Identity [displayName=thomas, uid=1254, email=java@thomas.epita, attributes=null]
```

Update Operation

The user required to enter old user id to modify user details in the database.

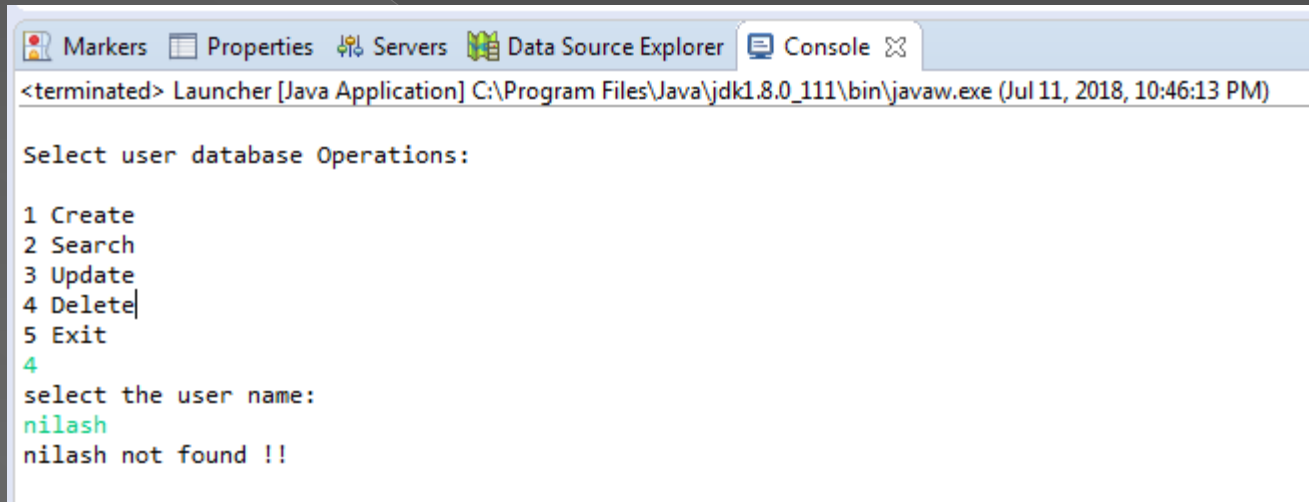


```
Markers Properties Servers Data Source Explorer Console
<terminated> Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:43:22 PM)
Select user database Operations:

1 Create
2 Search
3 Update
4 Delete
5 Exit
3
please input uid
1254
please input the new user name :
nilash
please input the new email
nilash789@epita.java
Updated nilashSucesfully
```

Delete Operations

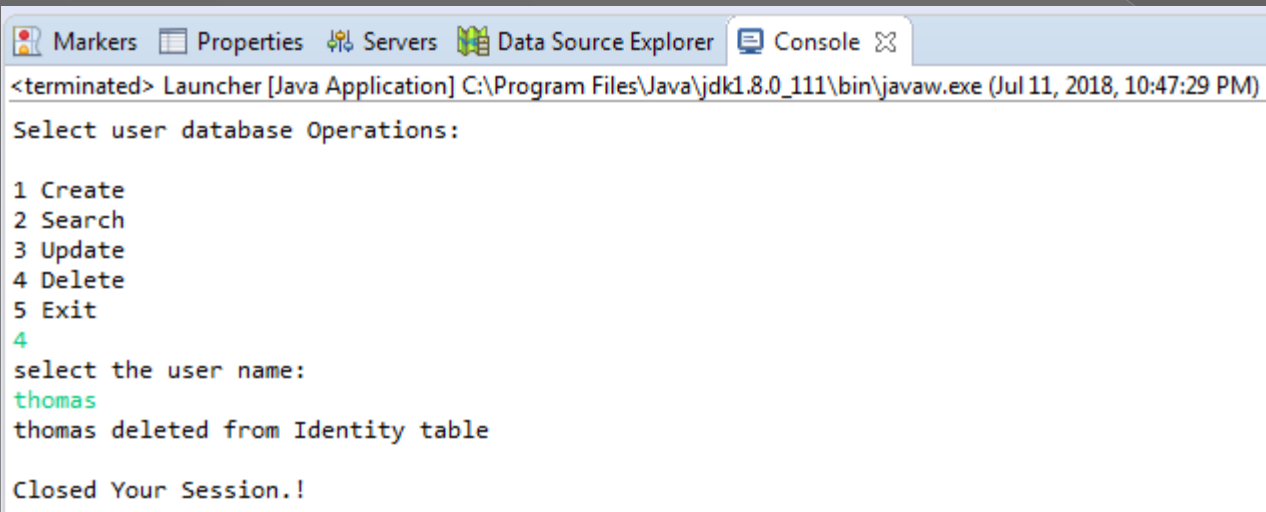
the user is required to input the Display Name to perform delete operation. Based on the input if the database has the value it will delete the entire details of that identity in the database.



```
Markers Properties Servers Data Source Explorer Console
<terminated> Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:46:13 PM)

Select user database Operations:

1 Create
2 Search
3 Update
4 Delete|
5 Exit
4
select the user name:
nilash
nilash not found !!
```



```
Markers Properties Servers Data Source Explorer Console
<terminated> Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:47:29 PM)

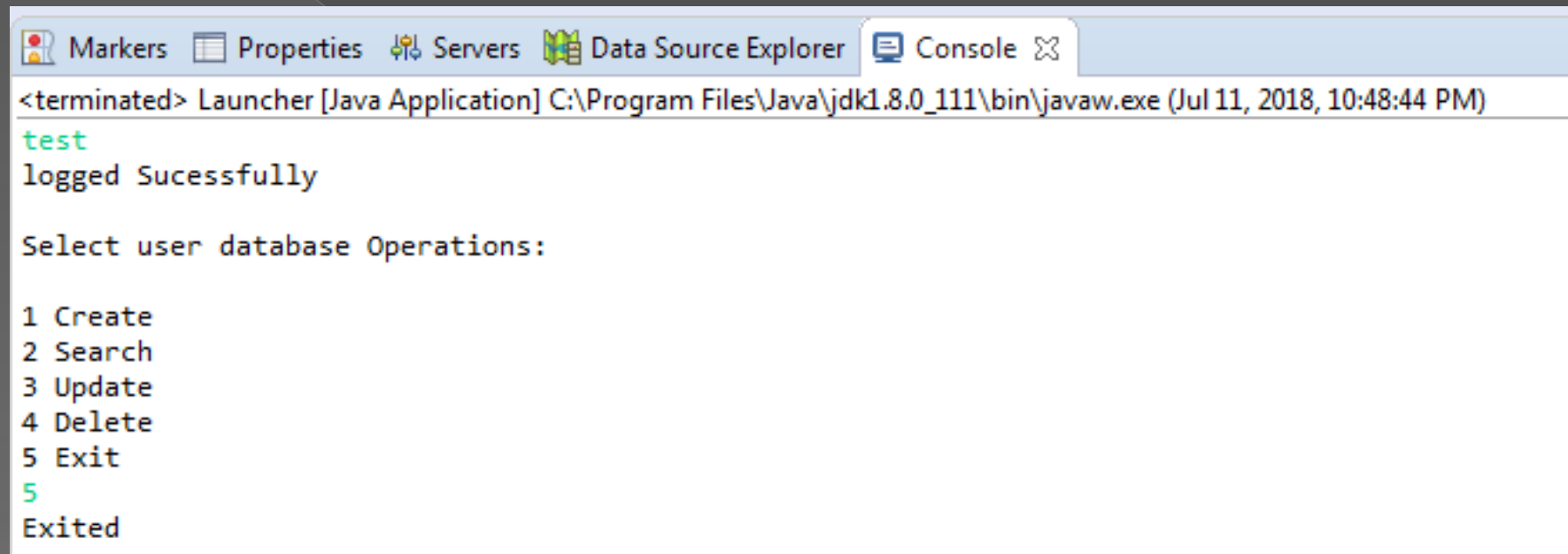
Select user database Operations:

1 Create
2 Search
3 Update
4 Delete
5 Exit
4
select the user name:
thomas
thomas deleted from Identity table

Closed Your Session.!
```

Exit

If the user wants to exit form the application. This is the area to exit.



The screenshot shows a Java application window with a console tab. The console output indicates a successful login for a user named 'test' and a menu of database operations. The user has selected option 5, 'Exit', and the application has responded with 'Exited'.

```
<terminated> Launcher [Java Application] C:\Program Files\Java\jdk1.8.0_111\bin\javaw.exe (Jul 11, 2018, 10:48:44 PM)
test
logged Sucessfully

Select user database Operations:

1 Create
2 Search
3 Update
4 Delete
5 Exit
5
Exited
```

Configuration instructions

Prerequisites

JDK (Java Development Kit)

Eclipse (IDE)

Derby (Database Server)

Java Configuration

<https://www.java.com/en/download/help/path.xml>

Installation Eclipse

http://www.eclipse.org/downloads/eclipse-packages/?show_instructions=TRUE

Derby Configuration

https://www.eclipse.org/datatools/project_connectivity/connectivity_doc/Connect%20to%20Derby%20using%20DTP%20M2.htm



Bibliography

<http://thomas-broussard.fr/work/java/courses/project/fundamental.xhtml>

<https://stackoverflow.com/>

<https://www.youtube.com/watch?v=h1rYIMrvNyE>

Java.end();

thomas-broussard.Thank you();

